Discussion Draft for the NIST Accordion Mode Workshop 2024

Proposal of Requirements for an Accordion Mode

Yu Long Chen
Michael Davidson
Morris Dworkin
Jinkeon Kang
John Kelsey
Yu Sasaki
Meltem Sönmez Turan
Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology

Donghoon Chang
Nicky Mouha
Strativia
Largo, MD

Alyssa Thompson
National Security Agency
Fort Meade, MD

April 2024

Table of Contents

1. Introduction

A block cipher mode of operation specifies a technique to process multiple blocks of data through an underlying block cipher. The National Institute of Standard and Technology (NIST) Special Publication (SP) 800-38 series [1–9] specifies several block cipher modes of operation (or "modes" for short). The modes are designed to address a variety of security requirements and use cases.

The NIST Crypto Publication Review Board has initiated a review of the SP 800-38 series; as a part of the review, some of the limitations of approved modes have been identified in (draft) NIST Internal Report (IR) 8459 [10]. To discuss how to address some of these limitations, NIST hosted the Third Workshop on Block Cipher Modes of Operation[1] in 2023.

NIST intends to develop a new block cipher mode of operation that is a tweakable, variable-input-length-strong pseudorandom permutation (VIL-SPRP). NIST introduces the term *accordion cipher mode* — or simply *accordion mode* — for the proposed mode because it would act as a cipher on a range of sizes for the message input. In order to facilitate the vetting of the accordion mode, NIST expects to require a reduction proof to the security of the underlying block cipher.

A well-designed accordion mode could provide security and performance advantages over the block cipher modes specified in the SP 800-38 series. For example, an accordion mode may provide additional features, better implementation and better security properties than AES-GCM, including, but not limited to, nonce-misuse resistance, support for short tags, nonce hiding, and key commitment, etc.

The aims of this document are to 1) establish terminology and notation for the development effort, 2) discuss the design requirements for an accordion mode, and 3) identify related topics for discussion at the upcoming NIST Accordion Mode Workshop 2024. In some cases, NIST offers preliminary proposals to prompt and focus the discussions. At a minimum, NIST would like to get sufficient feedback to decide appropriate parameter sizes. Public feedback will also be used to determine the next steps in the development effort.

This document describes three categories of applications for an accordion mode and for each category indicates how a derived function could satisfy the application. The choice and standardization of particular derived functions may be of independent interest, once an accordion mode is developed and approved. For the purposes of the workshop, NIST is mostly interested in how the derived functions should affect the design requirements or evaluation criteria.

In addition to feedback during the workshop, NIST welcomes written comments on the proposed development effort. Comments should be submitted to `ciphermodes@nist.gov` by July 1, 2024.

[1] https://csrc.nist.gov/Events/2023/third-workshop-on-block-cipher-modes-of-operation

1.1 Background

The term *enciphering mode* is used here to describe a block cipher mode of operation that provides length-preserving encryption for some set of supported input lengths. The term *tweakable enciphering mode* indicates that the enciphering mode would take an additional input called a *tweak*. We define an *accordion (cipher) mode* to be a tweakable enciphering mode that takes message input with variable input lengths (VIL) and behaves as a strong pseudorandom permutation (SPRP).

Some related design approaches are the Hasty Pudding Cipher [11], AEZ, [12] HCTR2 [13], Glevian and Vigornian [14]. There are also constructions based on other primitives such as XChacha and tweakable block ciphers: examples include Adiantum [15] and ZCZ [16]. Note that in order to use an accordion mode for authenticated encryption, we might apply (for example) the Encode-then-Encipher Encryption [17] technique proposed by Bellare and Rogaway. Another way to achieve variable block lengths is to use format-preserving encryption, but in this case the input domain is usually small. Note that NIST standards FF1 and FF3-1 [9], although AES based, are expected to be much slower than a typical accordion mode. In addition, FF3-1 was shown to be insecure [18].

While none of the above designs may meet the exact requirements to be outlined in this document, they provide valuable background and reference material, and they may be a good basis to begin thinking about a NIST accordion mode.

1.2 Overview

An accordion mode can be viewed as one layer of a larger process, see Figure 1. In particular, the underlying block cipher is the innermost layer, and this is called by the accordion mode. The accordion mode is called by a *derived function*, which defines how the accordion mode will be used, but does not do any cryptographic processing of inputs on its own. For example, a derived function could provide AEAD (authenticated encryption with associated data) functionality by encoding its inputs and providing them to the accordion mode. The derived function would then be called by an application, such as TLS. One motivation for the development of an accordion mode is that it can be used to support a variety of functionality which currently require their own enciphering modes.

The derived function layer will specify an encoding for the inputs to the accordion mode and the accordion mode will specify the inputs for the block cipher. The general flow and terminology for encryption are as follows. The outer layer takes in the raw data, or message, to be encrypted and applies an encoding, for example, by adding some integrity check bits. The shared secret or master key provided to the outer layer may be used directly as the encryption key for the accordion mode, or may be used to specify some new accordion key. Using any other inputs provided, the derived function layer defines a tweak for the accordion mode. The accordion tweak, encoded message and accordion key are used to call the accordion mode encryption function. Within the accordion mode encryption, a block cipher key, which may be the same as the accordion key, is used to call the block cipher. The encoded message blocks are input to the block cipher encryption and ciphertext blocks

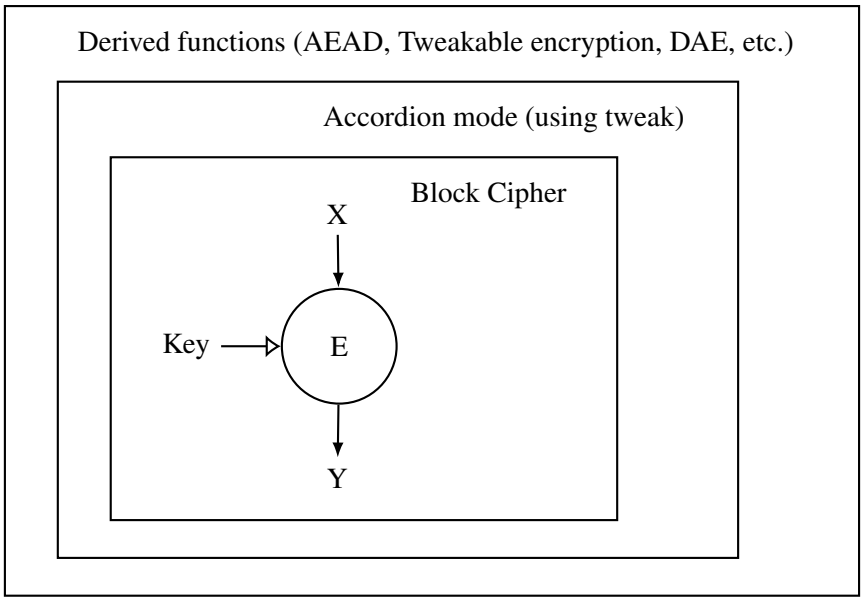

Fig. 1. Layered structure of the accordion mode and derived functions.

are returned. The accordion mode collects the ciphertext blocks into a single ciphertext, which is returned to the derived function.

When needed for specificity, we use the above terminology conventions to label the inputs at each layer. (i.e. *accordion key* and *encoded message* at the accordion mode layer, *block cipher key* at the block cipher layer, etc). When the layer is clear from the context, we drop the specificity and simply use "key," "message," and "tweak".

NIST is interested in standardizing an accordion mode that could support derived functions for at least three applications: AEAD, tweakable encryption, and deterministic authenticated encryption. In the following sections, we define the mode and derived functions in more detail, as well as proposed security goals and parameters to be supported.

2. The Accordion Mode

An accordion mode is an enciphering mode of a fixed-length block cipher that constructs a tweakable block cipher with a variable input length, typically allowing input sizes multiple of the block size of the underlying cipher. As discussed below, NIST believes that an accordion mode would be useful both in its own right, and as a component for supporting many applications that currently use some other block cipher chaining mode.

An accordion mode will have several parameters that are fixed within a given instance of the mode. The final standard may support multiple parameter sets. In Section 4, NIST proposes some guidance for selecting these parameters.

2.1 Notation

The following table provides the notation for elements of the accordion mode.

Symbol	Definition
a	Integer such that ag is the minimum allowed message size in bits
b	Integer such that bg is the maximum allowed message size in bits
C	Ciphertext
E	Underlying block cipher
g	Granularity
K	Secret key
k	Length of the secret key K in bits
ℓ	Length of the message M in bits
M	Message
n	Block size of block cipher E
T	Tweak
s	Length of tweak in bits
s_{min}	Minimum allowed bit size for tweak T
s_{max}	Maximum allowed bit size for tweak T

The Accordion mode consists of the encryption algorithm `A.enc` and the decryption algorithm `A.dec`. The three inputs to `A.enc` are a secret key $K \in \{0,1\}^k$, a tweak $T \in \{0,1\}^s$, where $s_{min} \leq s \leq s_{max}$, and a message $M \in \{0,1\}^\ell$ where $\ell \in \{ag, (a+1)g, \ldots, bg\}$; the output is a ciphertext $C \in \{0,1\}^\ell$:

$$\texttt{A.enc}(K,T,M) = C. \tag{1}$$

For any fixed values of K and T, `A.enc` is a permutation, and the decryption algorithm `A.dec` is its inverse, so that

$$\texttt{A.dec}(K,T,C) = M. \tag{2}$$

2.2 Security Goal

The security goal for an accordion mode is to behave as a tweakable VIL-SPRP [15, 19]. Informally, this means that, for a random key, any change to the encryption/decryption inputs (message/ciphertext and tweak) has a randomizing effect on the outputs (ciphertext/message).

A (q,σ,t)-distinguisher against the accordion is an algorithm $\mathscr{D}$ making at most q oracle queries with the total number of queried blocks being at most σ, running in time at most t, and outputting a single bit b. The following security game defines the security we expect from this construction:

1. At the beginning of the game, the challenger generates a random bit b and a random key K uniformly at random from the key space.

2. The distinguisher $\mathscr{D}$ is allowed to make up to q queries to the challenger, of the form encrypt(T,x) or decrypt(T,x):

   (a) If $b = 0$, the challenger answers these queries as follows:

   $$\text{encrypt}(T,x) = \texttt{A.enc}(K,T,x)$$
   $$\text{decrypt}(T,x) = \texttt{A.dec}(K,T,x).$$

   (b) If $b = 1$, then for each distinct choice of $(T,|x|)$, where $|x|$ denotes the length of x, the challenger selects and remembers a new random permutation on $2^{|x|}$ elements, $\Pi_{T,|x|}$. It answers queries using $\Pi_{T,|x|}, \Pi^{-1}_{T,|x|}$, as follows:

   $$\text{encrypt}(T,x) = \Pi_{T,|x|}(x)$$
   $$\text{decrypt}(T,x) = \Pi^{-1}_{T,|x|}(x).$$

3. After making q such queries, the distinguisher $\mathscr{D}$ must guess b.

The accordion mode is required to ensure that for any (q,σ,t)-distinguisher $\mathscr{D}$, the advantage of winning this game is negligibly small for acceptable values of q, σ, and t. NIST prefers standard model proofs when possible.

3. Derived Functions and Applications

This section describes derived functions of an accordion mode for the following three categories of applications: AEAD, tweakable encryption, and deterministic authenticated encryption. Each of these functions will have some application-specific inputs, along with the message and a key. The particular use case of the function motivates a set of properties that should be achieved. For example, AEAD by definition includes a means of data authentication, while tweakable encryption does not. Each subsection provides some guidelines for the derived function and an indication of how the function could be constructed out of an accordion mode.

3.1 Authenticated Encryption with Associated Data (AEAD)

Any derived function for AEAD authenticates the message and associated data. We define τ to be the bits of authentication security, meaning that $2^{-\tau}$ is the maximum allowable probability that any given invalid ciphertext will be accepted as valid.

One possible construction for an AEAD from an accordion mode is shown in Figure 2. The message is padded, both to provide τ fixed bits to be used for authentication, and to ensure that the input is an allowed size for the accordion mode. This construction takes a nonce and some associated data, which are encoded into the accordion tweak. This application motivates the accordion mode's support for variable-length, and relatively long, tweaks.

In addition, if the AEAD takes a nonce as one of the inputs, then nonce-misuse resistance is an important property. If the nonce is encoded as part of the tweak, then this property is provided by the security definition of the accordion mode, which ensures that reusing a tweak[2] imposes the smallest possible security loss. Specifically, encrypting the same input twice with the same tweak will result in the same output, but otherwise each new encryption will appear to be randomly selected from the set of not-yet-seen outputs of the correct length from the accordion mode.

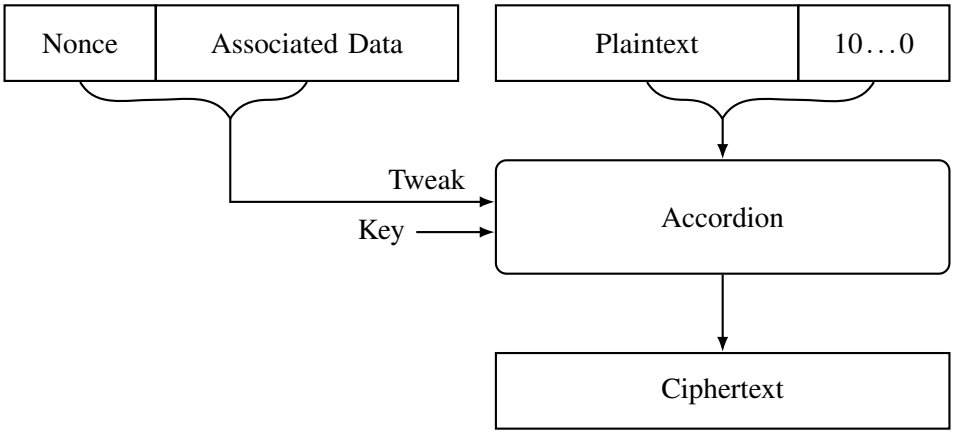

Fig. 2. Accordion-based authenticated encryption using an encode-then-encipher approach.

3.2 Tweakable Encryption

A tweakable encryption derived function based on an accordion mode will make use of the accordion tweak to encrypt a message. Unlike in the AEAD case, an integrity check is not encoded into the message, so message authentication is not provided. Although padding may still need to be applied to the message to reach an allowed message size, a small granularity could enable encryption without ciphertext expansion, making this derived function useful for storage encryption.

Figure 3 shows one possible approach to using accordion-based tweakable encryption for storage devices. Here, each data unit is encrypted with the secret key and a tweak encoding the data unit index. For the mode to be practical for this application, changing the tweak should be very efficient. In addition, keys or values derived from the key may end

[2]Note that the tweak is not a *nonce*, as it is permissible to use the same tweak multiple times.

up as part of the message, so key-dependent input (KDI) security [20–22] may be desired. This property is discussed further in Section 5.2.4.

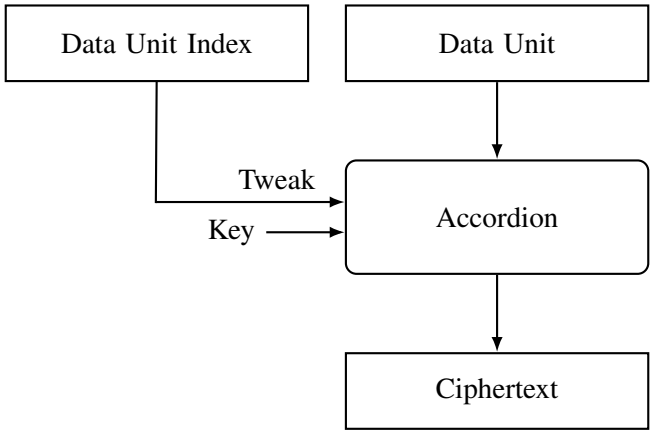

Fig. 3. Accordion-based tweakable encryption for storage devices.

3.3 Deterministic Authenticated Encryption (DAE)

A deterministic authenticated encryption derived function provides authentication without a tweak. A common use case for this derived function is key wrapping. One approach to using an accordion mode DAE to perform key wrapping is shown in Fig. 4.

As DAE does not involve a tweak, any constant minimal length tweak could be chosen to specify the mode. (In the diagram, the empty string is furnished as the tweak.)

In a DAE derived function, the input needs to be padded by τ constant bits to ensure τ-bit authentication, and then padded by any additional amount needed to get to an allowed input size. After decryption, the padding bits are verified for correctness to authenticate the message.

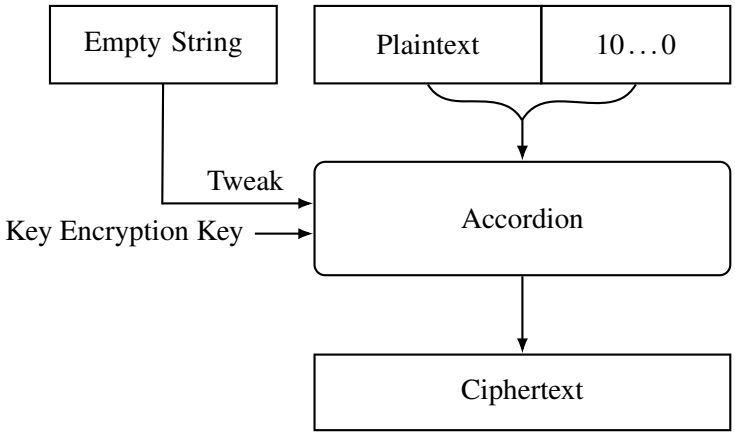

Fig. 4. Accordion-based DAE for key wrapping.

4. Requirements for Accordion Parameters

In this section, NIST gives preliminary thoughts and proposals about the ranges of sizes that should be required or supported for its components. An important goal of the workshop is to help NIST decide accordion mode requirements that balance a variety of needs, such as providing desirable new properties, promoting efficient implementation, and potentially replacing currently approved NIST modes in some applications.

4.1 Block Size of the Underlying Cipher

The accordion mode must support the AES block cipher [23], for which the block size is $n = 128$. NIST proposes that the accordion mode should also support 256-bit block ciphers, in case NIST approves one in the future. NIST is interested in feedback about how the additional block size might affect the design of the accordion mode.

4.2 Key Size

NIST proposes that the accordion mode must support 256-bit keys, with a targeted 256-bits of security in the single user setting. NIST is interested in feedback about whether to require or allow support for other key sizes and security strengths.

4.3 Tweak Size

NIST proposes that the accordion mode should support variable-length tweaks, with some minimum bit-length s_{min} up to some maximum bit-length s_{max}. For some applications, NIST expects that the accordion mode will need to support tweaks with a large maximum length. For example, in order to construct an AEAD derived function from an accordion mode efficiently, the tweak will likely contain the AEAD's associated data. NIST would like suggestions for an appropriate maximum tweak value, to support this use case, and an appropriate minimum tweak value in general.

Some applications, such as storage encryption, can work with a much shorter tweak, and could therefore benefit from an efficient method of processing a fixed length tweak, such as 128 bits, with a separate method for processing other tweak lengths. NIST is interested in feedback about the tradeoffs for this property.

4.4 Message Lengths

Given the granularity of the mode g, the minimum message bit-length ag, and the maximum message bit-length bg, the mode supports a total of $(b-a+1)$ allowed message lengths, $\ell \in \{ag, (a+1)g, \ldots, bg\}$. A mode with a granularity of one bit ($g = 1$) can process *any* message between its minimum and maximum length, whereas a scheme with granularity of 128 bits can only process messages whose length is a multiple of 128 bits between its minimum and maximum lengths.

In general, a mode becomes more useful the smaller its granularity and the larger its range of allowed message lengths. For example, a mode with $g = 8$ is more useful than one with $g = 128$, and a mode with $ag = 256, bg = 4096$ is less useful than one with $ag = 128, bg = 2^{50}$. On the other hand, it may be possible to make the mode more efficient or simpler by having a larger granularity or a narrower range of allowed lengths.

Applications that need non-expanding encryption of some particular size can't make use of an accordion mode that does not support that particular size. Most other applications can apply padding to the input to the accordion mode.

NIST seeks suggestions about what message lengths should be supported.

5. Desirable Properties of an Accordion Mode

Along with the proposed parameters listed above, a number of other properties would be desirable or may be useful in some contexts.

5.1 Performance Targets

While performance is not as critical as security, the better the performance of the mode, the more useful it is likely to be. The most important platform for the mode is likely to be relatively powerful processors in desktop or laptop computers and those used in cloud environments, with hardware AES support. Performance in dedicated hardware and on constrained devices is also worthwhile but is not NIST's focus for the accordion mode.

The security requirements for an accordion mode require making multiple passes over the message. Thus, it is almost certain to be slower than many existing one-pass block cipher modes like AES-GCM. However, the mode will ideally not be a lot more expensive than twice the cost (in time, block cipher calls, gates, etc.) of AES-GCM over the same input size.

The mode should allow substantial parallelism for large input sizes. Modern CPUs often support doing multiple AES instructions or other operations in parallel, and the mode should allow the user to take advantage of this.

The mode should not impose too much additional overhead on small input sizes. A mode that does a substantial amount of setup work before processing an input block will generally perform poorly on small inputs. Some applications require efficient processing of even relatively short messages.

In most applications of an accordion mode, the tweak is likely to change often, while the key and input length will change less frequently. For some applications (for example, storage encryption), being able to change tweaks efficiently is critical. Thus, the mode should make changing the tweak, T, as efficient as possible.

NIST is interested in feedback on performance requirements for the accordion mode.

5.2 Additional Security Properties

The security definition proposed in Section 2 (i.e., VIL-SPRP in the single user setting) automatically promises some important security properties. In the following sections, several additional security properties that may be desired of the accordion mode or its derived functions are summarized.

5.2.1 Multi-User Security

In the multi-user setting, one of the main concerns is how the security of each functionality is affected as the number of users increases. For example, Luykx et al. proved that GCM, as a mode, does not have multi-user degradation [24]. NIST is interested in feedback on requirements related to the multi-user security of the accordion mode.

5.2.2 Beyond Birthday-Bound Security

Some encryption modes provide beyond birthday-bound (BBB) security, meaning the advantage of winning the game defined in Sect. 2.2 against any (q, σ, t)-distinguisher $\mathscr{D}$ is negligibly small even when more than $2^{n/2}$ blocks of input are processed–that is, $q > 2^{n/2}$ or $\sigma > 2^{n/2}$.

It is possible that NIST will standardize a 256-bit block cipher in the future, which may reduce the need for this property. However, initially the accordion mode must support AES, with a 128-bit block. NIST would like feedback on any immediate needs for beyond-birthday-bound security and what usage bounds applications may require.

5.2.3 Key and Context Commitment

Key and context commitment are properties that ensure that the same ciphertext cannot be decrypted to different valid plaintexts based on the key or other context. These properties don't apply to an accordion mode, since it has no way of checking message validity. For *any* K_1, T_1, K_2, T_2, C the ciphertext C will decrypt to two different plaintexts: $P_1 = \texttt{A.dec}(K_1, T_1, C)$ and $P_2 = \texttt{A.dec}(K_2, T_2, C)$.

However, the properties of the accordion mode determine whether an authenticated encryption derived function built on the accordion mode can provide key commitment or context commitment. NIST is interested in what properties of an accordion mode are needed to support these security properties.

5.2.4 Key-Dependent-Input Security

Some applications involve encrypting inputs that include the key, or are derived from the key in some way. For example, when the operating system swaps the contents of some memory page to disk, the storage encryption mode being used could end up encrypting a copy of its own key. Most chaining modes' security proofs do not cover this use case. The property required to ensure security in this case is called *security against key-dependent*

inputs (KDI security). As shown by Halevi and Krawczyk in ACM-CCS 2007 [22], it is impossible to construct a KDI-secure deterministic construction such as the accordion mode if there is no restriction on generating KDIs.

NIST is interested in properties of the accordion mode that can help support KDI security for some useful subsets of possible key-dependent messages.

5.2.5 Nonce Hiding

Some applications may have a concern about privacy leakage occurring through public information, such as private information being included as part of the nonce or tweak. In these cases, the information should be encrypted along with the message, refer to [25] for more details about nonce-hiding. NIST would like feedback on any needs for this property and how it might affect the design of the accordion mode and its associated derived functions.

6. Next Steps

NIST plans to host a workshop in June 2024 to discuss requirements for the accordion mode in more detail. Although a process for selecting an accordion mode to standardize has not yet been determined, NIST intends to use the feedback and discussions from the workshop to determine what the next steps should be. Following the workshop, NIST is likely to establish a collaborative process for developing a new accordion mode. This may include soliciting proposals that meet these requirements and holding further workshops to discuss or develop different aspects of accordion modes under consideration. NIST intends to eventually select one or more accordion modes (along with the surrounding derived functions needed to support important applications) for standardization.

The `ciphermodes-forum@list.nist.gov` emailing list has been established for dialogue regarding NIST's Block Cipher Modes project. To subscribe to the mailing list, visit the following page https://groups.google.com/a/list.nist.gov/g/ciphermodes-forum.

References

[1] Dworkin M (2001) Recommendation for Block Cipher Modes of Operation: Methods and Techniques, NIST SP 800-38A. https://doi.org/10.6028/NIST.SP.800-38A.

[2] Dworkin M (2010) Recommendation for Block Cipher Modes of Operation: Three Variants of Ciphertext Stealing for CBC Mode, Addendum to NIST SP 800-38A. https://doi.org/10.6028/NIST.SP.800-38A-Add.

[3] Dworkin M (2005) Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, NIST SP 800-38B. https://doi.org/10.6028/NIST.SP.800-38B.

[4] Dworkin M (2004) Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality, NIST SP 800-38C. https://doi.org/10.6028/NIST.SP.800-38C.

[5] Dworkin M (2007) Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, NIST SP 800-38D. https://doi.org/10.6028/NIST.SP.800-38D.

[6] Dworkin M (2010) Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices, NIST SP 800-38E. https://doi.org/10.6028/NIST.SP.800-38E.

[7] Dworkin M (2012) Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping, NIST SP 800-38F. https://doi.org/10.6028/NIST.SP.800-38F.

[8] Dworkin M (2016) Recommendation for Block Cipher Modes of Operation: Methods for Format-Preserving Encryption, NIST SP 800-38G. https://doi.org/10.6028/NIST.SP.800-38G.

[9] Dworkin M (2019) Recommendation for Block Cipher Modes of Operation: Methods for Format-Preserving Encryption, Draft NIST SP 800-38G Revision 1. ttps://doi.org/10.6028/NIST.SP.800-38Gr1-draft.

[10] Mouha N, Dworkin M (2023) Report on the Block Cipher Modes of Operation in the NIST SP 800-38 Series (initial public draft), NISTIR 8459. https://doi.org/10.6028/NIST.IR.8459.ipd.

[11] Schroeppel R (1998) Hasty Pudding Cipher Specification. Available at http://richard.schroeppel.name:8015/hpc/hpc-spec.

[12] Viet Tung Hoang PR Ted Krovetz (2017) AEZ v5: Authenticated encryption by enciphering. https://www.cs.ucdavis.edu/~rogaway/aez/aez.pdf.

[13] Wang P, Feng D, Wu W (2005) HCTR: A Variable-Input-Length Enciphering Mode. *Information Security and Cryptology, First SKLOIS Conference, CISC 2005, Beijing, China, December 15-17, 2005, Proceedings*, eds Feng D, Lin D, Yung M (Springer), *Lecture Notes in Computer Science*, Vol. 3822, pp 175–188. https://doi.org/10.1007/11599548_15

[14] Campbell P (2023) GLEVIAN and VIGORNIAN: Robust beyond-birthday AEAD modes, Cryptology ePrint Archive, Paper 2023/1379. https://eprint.iacr.org/2023/1379 Available at https://eprint.iacr.org/2023/1379.

[15] Crowley P, Biggers E (2018) Adiantum: length-preserving encryption for entry-level processors. *IACR Trans Symmetric Cryptol* 2018(4):39–61. https://doi.org/10.13154/tosc.v2018.i4.39-61

[16] Bhaumik R, List E, Nandi M (2018) ZCZ - Achieving n-bit SPRP Security with a Minimal Number of Tweakable-Block-Cipher Calls. *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part I*, eds Peyrin T, Galbraith SD (Springer), *Lecture Notes in Computer Science*, Vol. 11272, pp 336–366. https://doi.org/10.1007/978-3-030-03326-2_12

[17] Bellare M, Rogaway P (2000) Encode-Then-Encipher Encryption: How to Exploit Nonces or Redundancy in Plaintexts for Efficient Cryptography. *Advances in Cryptology - ASIACRYPT 2000, 6th International Conference on the Theory and Application of Cryptology and Information Security, Kyoto, Japan, December 3-7, 2000, Proceedings*, ed Okamoto T (Springer), *Lecture Notes in Computer Science*, Vol. 1976, pp 317–330. https://doi.org/10.1007/3-540-44448-3_24

[18] Beyne T (2021) Linear cryptanalysis of FF3-1 and FEA. *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part I*, eds Malkin T, Peikert C (Springer), *Lecture Notes in Computer Science*, Vol. 12825, pp 41–69. https://doi.org/10.1007/978-3-030-84242-0_3. Available at https://doi.org/10.1007/978-3-030-84242-0_3

[19] Halevi S, Rogaway P (2003) A Tweakable Enciphering Mode. *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, ed Boneh D (Springer), *Lecture Notes in Computer Science*, Vol. 2729, pp 482–499. https://doi.org/10.1007/978-3-540-45146-4_28

[20] Ball MV (2008) NIST's consideration of XTS-AES as standardized by IEEE Std 1619-2007, https://csrc.nist.gov/CSRC/media/Projects/Block-Cipher-Techniques/documents/BCM/Comments/XTS/XTS_comments-Ball.pdf.

[21] Black J, Rogaway P, Shrimpton T (2002) Encryption-Scheme Security in the Presence of Key-Dependent Messages. *Selected Areas in Cryptography, 9th Annual International Workshop, SAC 2002, St. John's, Newfoundland, Canada, August 15-16, 2002. Revised Papers*, eds Nyberg K, Heys HM (Springer), *Lecture Notes in Computer Science*, Vol. 2595, pp 62–75. https://doi.org/10.1007/3-540-36492-7_6

[22] Halevi S, Krawczyk H (2007) Security under key-dependent inputs. *Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007*, eds Ning P, di Vimercati SDC, Syverson PF (ACM), , pp 466–475. https://doi.org/10.1145/1315245.1315303. Available at https://doi.org/10.1145/1315245.1315303

[23] National Institute of Standards and Technology (2001) Advanced Encryption Standard (AES), FIPS 197. https://doi.org/10.6028/NIST.FIPS.197.

[24] Luykx A, Mennink B, Paterson KG (2017) Analyzing multi-key security degradation. *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II*, eds Takagi T, Peyrin T (Springer), *Lecture Notes in Computer Science*, Vol. 10625, pp 575–605. https://doi.org/10.1007/978-3-319-70697-9_20

[25] Bellare M, Ng R, Tackmann B (2019) Nonces are noticed: AEAD revisited. *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part I*, eds Boldyreva A, Micciancio D (Springer), *Lecture Notes in Computer Science*, Vol. 11692, pp 235–265. https://doi.org/10.1007/978-3-030-26948-7_9. Available at https://doi.org/10.1007/978-3-030-26948-7_9